Information Retrieval
CS 6900

Lecture 13

Razvan C. Bunescu

School of Electrical Engineering and Computer Science

bunescu@ohio.edu

Web Search Interfaces

- Web search engines need a **web-based interface**.
- Search page accepts a query string and submits it within an HTML `<form>`.
- Program on the server processes requests and generates HTML text for the top ranked documents with links to the original web pages. HTML response may also include:
  - Links to cached version.
  - Representative text snippet with keywords highlighted.
  - Targeted adds.
- Server program also allows for requests for more relevant documents for the current query.

Skeleton Code for Web Search Interface

http://ace.cs.ohio.edu/~razvan/courses/ir6900/server.tar.gz

- Based on model parameter, selects the IR model.
- Processes the query with IR model to get ranked results.
- Writes out HTML ordered list of 10 results starting at the rank of the start parameter.
  - Each item includes:
    - Link to the original URL, saved by the spider.
    - Anchor text is the page <title> extracted from file.
    - Additional link to local cached file.
- If all documents not already shown, creates a submit form for "More Results", starting from the next ranked item.
  - Needs session tracking, because HTTP is a "stateless" protocol.
    - Common solutions: Cookies and URL-rewriting.

HTML Forms

- HTML supports various types of program input in forms, including:
  - Text boxes.
  - Menus.
  - Lists.
  - Check boxes.
  - Radio buttons.
- When user submits a form, string values for all defined parameters are sent to the server program for processing.
- Server program uses these values to compute an appropriate HTML response page.

Simple Search Engine Form

```
<form action="http://localhost:8082" method="POST">
<p> <b> Enter your query: </b>
    <input type="text" name="query" size=40>
<p> <b>IR Model: </b>
   <select name="model">
   <option selected value="tf.idf"> Vector Space Model
   <option value="bm25"> Okapi BM25
   <option value="bool"> Boolean Model
   <option value="lm"> Language Model
  </select>
<p> <b>Use Relevance Feedback: </b>
<input type="checkbox" name="feedback" value="1">
<br> <br>
<input type="submit" value="Submit Query">
<input type="reset" value="Reset Form">
</form>
```

Simple Search Engine Form

```
<form action="http://localhost:8082" method="POST">
<p> <b> Enter your query: </b>
    <input type="text" name="query" size=40>
<p> <b>IR Model: </b>
   <select name="model">
   <option selected value="tf.idf"> Vector Space Model
   <option value="bm25"> Okapi BM25
   <option value="bool"> Boolean Model
   <option value="lm"> Language Model
  </select>
<p> <b>Use Relevance Feedback: </b>
<input type="checkbox" name="feedback" value="1">
<br> <br>
<input type="submit" value="Submit Query">
<input type="reset" value="Reset Form">
</form>
```

HTTP server program listening on this port

Hypertext Transfer Protocol (HTTP)

- Communication between client (e.g. user) and server (e.g. search engine) is done through HTTP.
- Two main methods for sending form data i.e. a *query string*:
  - GET:
    - The query string is simply appended to the URL of the program when the client issues the request to the server.
    - This query string can then be accessed by using the environment variable QUERY_STRING.
  - POST:
    - The server program reads the query string from standard input.
    - The query length can be unlimited.

http://oreilly.com/openbook/cgi/ch04_02.html

Building an HTTP Server

- **Python**:
  1. Modules BaseHTTPServer and cgi.
  2. Web application frameworks: Django, CherryPy, Web2Py, ...
     - Django is used by Instagram, Mozilla, Pinterest, PBS, ...
- **Ruby on Rails**:
  - Used by Github, Yammer, Scribd, Groupon, Shopify, ...
- **Java Servlets & JSP**:
  - Used by Google+, ...
- **PhP**:
  - Built-in capabilities.
  - Web application frameworks: CakePHP, Symfony.
- **ASP.NET**

Basic HTTP Request Handler

```
from BaseHTTPServer import BaseHTTPRequestHandler, HTTPServer

class myHandler(BaseHTTPRequestHandler):
  def do_GET(self):
    if self.path == "/":
        self.path = "/index.html”
    if self.path.endswith(".html"):
        f = open(curdir + sep + self.path)
        self.send_response(200)
        self.send_header('Content-type’, 'text/html’)
        self.end_headers()
        self.wfile.write(f.read())
```

Basic HTTP Request Handler

```
class myHandler(BaseHTTPRequestHandler):
    def do_POST(self):
        env = {'REQUEST_METHOD':'POST',
               'CONTENT_TYPE':self.headers['Content-Type']}
        form = cgi.FieldStorage(self.rfile, self.headers, env)
        self.send_response(200)
        self.end_headers()
        self.wfile.write("Query is " + form["query"].value)
```

Replace with HTML containing results from search engine.

Basic HTTP Request Handler

```
# Main entry point.
try:
  # Create a web server and define a handler to manage the HTTP request.
  server = HTTPServer((’localhost', 8082), myHandler)
  print 'Started HTTP server on port', str(8082)
  # Wait forever for incoming http requests.
  server.serve_forever()

except KeyboardInterrupt:
  print ’Shutting down the web server.'
  server.socket.close()
```

Launching the HTTP Server

```
[razvan@texas server]$ ls
cached/  SearchEngine.py  search.html  Server.py

[razvan@texas server]$ ./Server.py
Started httpserver on port 8082
localhost.localdomain -- [16/Nov/2013 17:36:57] "POST / HTTP/ ...
Query is: Ohio University
Start is: 0
localhost.localdomain -- [16/Nov/2013 17:37:02] "POST / HTTP/ ...
Query is: Ohio University
Start is: 10
```

Search Interface Refinements

- Highlight search terms in the displayed document.
  - Provided in cached file on Google.
- Allow for “advanced” search:
  - Phrasal search (“..”)
  - Mandatory terms (+)
  - Negated term (-)
  - Language preference
  - Reverse links.
  - Date preference.
  - Clustering of results.
- Machine translation of pages.

User Behavior

- Users tend to enter short queries:
  - Study in 1998 gave average length of 2.35 words.
- Users tend not to use advanced search options.
- Users need to be instructed on using more sophisticated queries.